NMSJ: A FILTERING TECHNIQUE FOR N-WAY JOINS IN WIRELESS SENSOR NETWORKS

Boubekeur DJAIL[1], Walid-Khaled HIDOUCI[2], Malik LOUDINI[3]

Join queries are widely used in wireless sensor networks. However, they engender high energy consumption, particularly for joins between a number of tables more than two: the n-way joins. The challenge is then to perform such queries while reducing consumed energy. Many techniques were proposed in this way, but most of them addressed only binary joins. N-way joins were rarely treated. With n-way joins, the consumed energy is excessively high, and the number of execution orders grows exponentially with the number of considered tables.

We present in this paper, a filtering technique to perform n-way join queries in wireless sensor networks. We try to significantly reduce the used energy, by adopting semi-join approach to filter out non-joinable tuples. We evaluate the performance of our technique by a comparison with the technique: Sens-join proposed par Stern and al. Obtained results show that our solution significantly outperforms Sens-join.

Keywords: wireless sensor networks, communication cost, in-network join, n-way join

1. Introduction

Wireless sensors are components that have limited memory and processing capabilities. They can be scattered in larges regions and communicate to each other with wireless support, to constitute a wireless sensor network. Each sensor picks up data to record in local data table. The set of sensors tables forms a distributed database table [14]. This table can be asked by relational queries, such: selections, projections, joins, …

Joins queries are very used in wireless sensor networks to recuperate data from several tables of the same distributed database table. However, running these queries results in high power consumption, which inevitably leads to the dysfunction of the sensor nodes, then the complete paralysis of the network. To reduce this energy consumption, researchers tent to reduce data transmission quantity because data transmission consumes more energy than data processing at nodes [16].

[1] ESI, Algeria, e-mail: b_djail@esi.dz
[2] ESI, Algeria, e-mail: wk.hidouci@gmail.com
[3] ESI, Algeria, e-mail: m_loudini@esi.dz

On the other hand, most of proposed techniques for joins queries in wireless sensor networks addressed binary joins. N-way joins were seldom treated.N-way joins queries must introduce more energy consumption than binary joins. Additionally, the execution order of such queries grows exponentially with the number of tables. Therefore, the challenge is to provide the best execution order and adopt the best solutions to reduce the overall energy consumed.

We present in this paper an efficient-energy technique based on semi-join principle to perform n-way join queries in wireless sensor networks. The rest of the paper is organized as follows: section 2 provides an overview of joins queries in wireless sensor networks. Related work is presented in Section 3. Section 4 gives more details of the proposed technique. Section 5 explains performed tests and presents a discussion about the obtained results. Finally, section 6 concludes the paper.

2. Joins queries in wireless sensor networks: an overview

2.1. Definitions.

A join query is performed at least between two tables. The result of its execution is a new table with tuples determined by concatenation of the tuples of each participate table. These tuples must verify a specified condition: the join predicate.

An equi-join is the query that has only equality operators in its join predicate. A binary join is performed between two tables. An n-way join is executed between three or more tables.

2.2. Implementation of join queries in wireless sensor networks

There are mainly two possible implementations in wireless sensor networks [7]:

External join: which executes the join query at the sink. The tuples must be transferred initially from nodes to the sink, i.e. before performing query. This implementation causes high energy consumption, given the important number of transmitted messages.

In-network join: that performs join query at a node or a set of nodes in the network. The number of transmitted messages is then decreased and the performance heighted.

2.3. Join types in wireless sensor network

There are two types of joins queries in wireless sensor networks [7].

- ‘Unique region' joins: There are joins which are performed in a single region, between the sensors of the same zone.
- ‘Inter-region' joins: These joins execute operations between at least two regions. Each region contains a distributed table.

This, if we consider the spatial aspect of the joins queries, but if we study the temporal aspect, there are three classes of joins queries [7]:

- ‘One shot’ joins: These joins are performed between static tables. Each table corresponds to a fixed window defined for a limited range of time or of tuples.
- ‘continuous' joins: There are the joins executed continuously by using sliding windows to delimit set of tuples that are concerned by join operation at each step.
- ‘periodic' joins: These joins can be considered as a particular case of continuous joins because they are executed repetitively at interval of times.

3. Related works.

Many techniques were proposed to address joins queries in wireless sensor networks. The first ones treated this query type without consider filtering to eliminate non-joinable tuples. Currently, the mainly approach is to adopt filtering principles to reduce considerably the messages transmitted number.

For the non-filtering techniques, we will cite followings works: Yao and Gehrke [14] studied the difference of the transmission cost between an external join and an in-network join. The result is that in-network execution consumes low energy for low join selectivity. Bonfils and Bonnet [1] searched the optimal node of an in-network join execution. They concluded that the optimal node is on the shortest path between the two nodes concerned by the join query, and this is the node which has more data. Coman and al. [2] proposed local join and mediated join techniques to address an inter-region join. Local join performs at one of the two regions concerned by the join whereas Mediated join executes the join at a zone situated between the two regions. Coman and al. concluded that no specific technique takes the best results all queries.

As to the techniques with filtering, Coman and al. [2] adopted semi-join principle and suggested Local Semi-Join technique which performs the join query in one of the two concerned regions. Yu and al. [15] presented Synopsis Join for treating an inter-region equi-join query between statics tables. They adopted a distributed alternate of the semi-join method. Min and al [10] considered various plans to execute a join query and they proposed a cost model to select the optimal plan under various conditions.

Specific joins queries were also addressed par different researchers. Mo and al. [11] studied spatial queries in wireless sensor network. Kang and al [6] treated iceberg join queries, where only tuples whose cardinality surpasses a given threshold are admitted to the join operation.

Those described techniques addressed mainly binary join. Solutions to treat n-way joins were too rare.

Stern and al. [10] proposed SENS-join, a technique that consider n-way join queries. SENS-join performs join queries at sink by using filters determined at root nodes and based on the relevant records.
SENS-join runs in five steps:
- The join query is diffused by the sink to the root nodes of different regions.
- Join attributes are transmitted by the root nodes to the station base. These attributes serve to determine the join query filter.
- The generated filter is determined and then diffused to the root nodes.
- Based on the filter, root nodes perform the semi-join with their tuples. So, the result of each operation is communicated to the base station.
- Finally, at the sink, the final join query is performed.

This technique introduces a high quantity of transmitted messages from nodes to the sink.

In the same perspective, we proposed a non-filtering technique NLJ (N-way Local Join) [4] to treat n-way join queries.
NLJ technique runs in three steps:
- The query is transmitted by the sink to the root nodes of concerned regions.
- The join query is executed as a succession of binary joins between each region with its neighbour. The order of these executions is determined based of the left linear trees technique[12].
- The last obtained result, at the last region, is communicated to the sink.

We also suggested a filtering technique NLSJ (N-way Local SemiJoin) [3] for the same joins type, which is based on semi-join principle.
NLSJ runs in three steps:
- The query is communicated to the root nodes of the regions.
- An intermediate join is performed for each nodes' pair (S_i, S_{i+1}). S_{i+1} is the site chosen as the nearest site to Si. This execution is done as follow:
  - The S_{i+1} site transmits the join attribute to S_i.
  - At S_i, a semi-join is executed between the join attribute and the local table.
  - The determined result is communicated to S_{i+1}.
  - At S_{i+1}, the complete join query is performed.
- The result of the last intermediate join, which corresponds to the final result of the join query, is delivered to the base station.

4. N-way Mediated Semi-Join (NMSJ).

4.1. NMSJ description

In this paper, we present N-way Mediated Semi-Join (NMSJ) for n-way join queries in wireless sensor networks. We consider one-shot inter-region joins with syntax as:

```
SELECT <List of attributes from T1, T2, ... and Tn>
FROM  T1, T2, …, Tn
WHERE  pred(T1)  AND pred(T2)  … AND pred(Tn)
AND join-expression (T1.join-attribut , T2.join-attribut ,…,
Tn.join-attribut)
```

With:
T1, T2, …, Tn : Static tables which participate to the query.
Pred (Ti) : is the selection predicate of a relation Ti.
Join-expression: is the join condition expression.

This join query type is very used by applications that need to join data from several tables in wireless sensor networks, such the application for the control traffic vehicle, and the application for monitoring birds' immigration.
For this last example, we can write the following query:

```
SELECT    B1.BId, B1.time, B2.time, B3.time
FROM       B1, B2, B3
WHERE    (B1.time IN rg1) AND (B2.time IN rg2) AND (B3.time IN rg3)
AND (B1.BId = B2. BId) AND (B2. BId= B3. BId)
```

Where:
B1, B2, B3: represent the tables in different regions.
Id, time: are the attributes in each table.
rg1, rg2, rg3: define the times intervals during which birds pass respectively through the three regions.

N-way Mediated Semi-Join (NMSJ) is an improvement of NLSJ [3] that we proposed for the same join query type. Additionally, to the in-network execution principle, NMSJ adopts the Semi-Join method to reduce non-joinable tuples before performing the query. It also uses the left linear trees technique [12] to decide the choice of the execution order.

NMSJ executes an n-way join query in three phases:
Phase 1. Query dissemination.

The sink diffuses the query in the network, to root nodes of all concerned regions. For this, GPSR protocol [8] is used to guarantee a correct receipt of the query.

Additionally, the nodes of each region send their tuples to the root node, by using GPS or localization algorithms [12] to determine their positions and those of their neighbours.

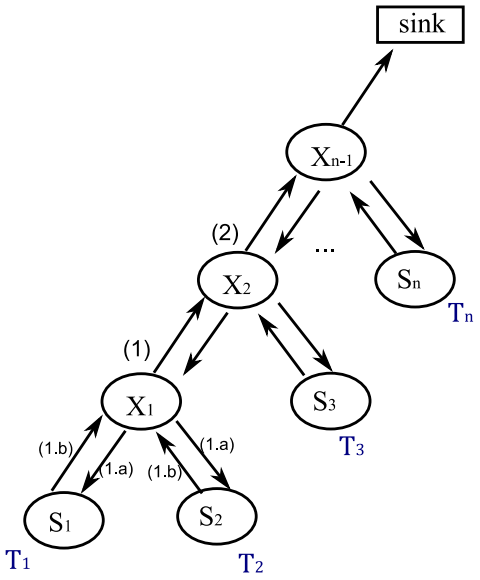

Fig.1 - N-way Mediated Join (NMSJ) principle.

Phase 2. Query execution

The join query is performed by several steps. At each step, an intermediate join is executed by a pair of nodes (S_i, S_{i+1}). S_{i+1} is selected as the nearest node to the currently site S_i.

Each intermediate join is realized in five times (Fig.1):

i. The nodes (S_i, S_{i+1}) select the site X_i where the intermediate-join must be performed. X_i is designated from the two sites (S_i, S_{i+1}) as the site with the low tuples number to participate in the join. The goal is to reduce the tuples number to transmit.
ii. The join attribute of the X_i site is transmitted to the second site (non-chosen site).
iii. A semi-join between the join attribute of X_i and the second relation is performed.
iv. The result of semi-join operation is communicated to X_i.
v. The intermediate join is finalized at this site.

Phase 3. The result at the last site is delivered to the sink.

4.2. Illustrative example

To better explicit the NMSJ principle, we present the following example, where we execute a join query between three tables.

The first intermediate join is performed by the nodes pair (S1, S2). The site S1 has the low number of tuples to transmit, so it is selected as X1 to finalize the intermediate join. X1 (S1) send the attribute-join values to S2, which realizes

the semi-join with its tuples. The result is then communicated to X1 where the final result of T1 join T2 is determinate (Fig. 2 (a)).

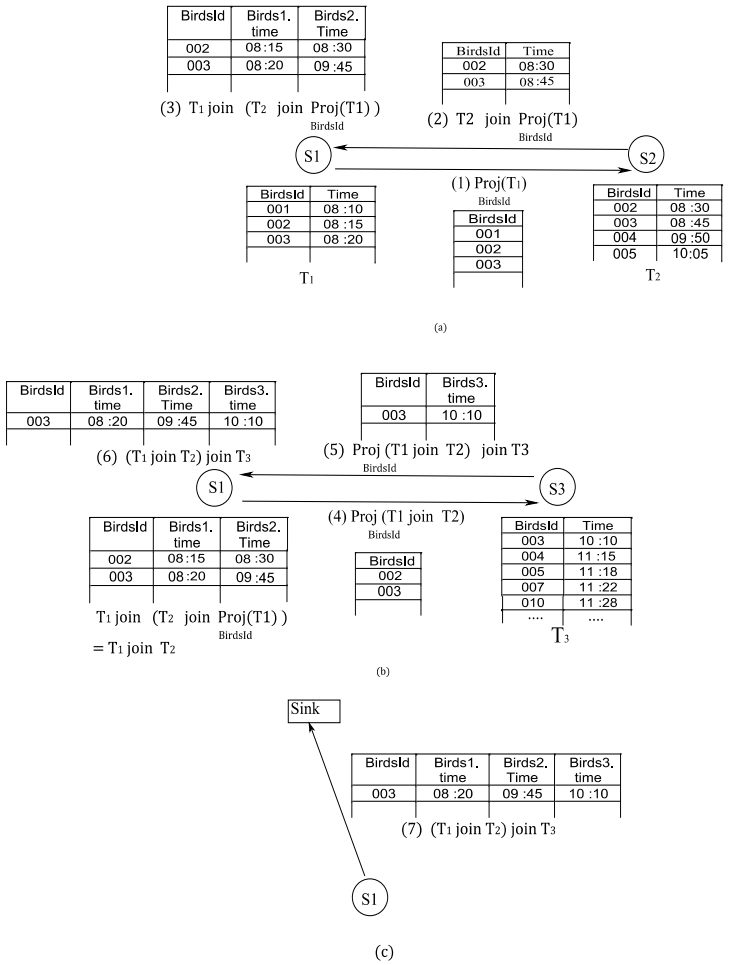

Fig.2 - An example for N-way Mediated Semi-Join (NMSJ) execution.

After this first step, the intermediate join between the nodes pair (S2, S3) is performed in the same manner as previously (Fig. 2 (b)). At end, the final result is delivered to the sink (Fig. 2 (c)).

4.3. Cost calculation

We present in this follow the formula that we have determined to calculate the cost of NMSJ technique.

This cost is represented by the number of messages transmitted during a query execution.

For n tables, the cost is done by:

$$\left(\frac{5n-1}{4}\right)T + \frac{1}{4}f_1T^2 + \frac{1}{4}f_1f_2T^3 + \cdots + \frac{1}{4}f_1f_2 \ldots f_{n-1}T^n + f_1f_2 \ldots f_{n-1}T^n \quad (1)$$

If we assume that all joins selectivity are equals, we obtain the following formula :

$$\left(\frac{5n-1}{4}\right)T + \frac{1}{4}fT^2 + \frac{1}{4}f^2T^3 + \cdots + \frac{1}{4}f^{n-1}T^n + f^{n-1}T^n \quad (2)$$

where:

$\frac{5n-1}{4} = nT + (n-1)\frac{T}{4}$ is the first cost transmission before the first intermediate join.

$\frac{1}{4}f\,T^2$ is the cost transmission of the result of the first intermediate-join.

$\frac{1}{4}f^2T^3$ is the cost transmission of the result of the second intermediate-join.

$\frac{1}{4}f^iT^{i+1}$ is the cost transmission of the result of the i^{th} intermediate-join.

5. Performance analysis

5.1. Experimentation environment

We perform NMSJ simulation on the NS3 simulator. We assume in this simulation the following:

- Each region is structured in arborescence, with one node designated as root.
- The table size is 2000 tuples.
- The tuple size is 40 bytes.
- The message size is the same than the tuple size.
- The column size is 10 bytes.
- The result tuple size is 30 bytes.

The simulation was realized by considering three tables and also five tables. In each case, we determine communication cost by measuring the number of transmitted messages according to selectivity factors values of intermediates joins.

These values are designated from the $[10^{-5}, 10^{-4}]$ range and from $[10^{-4}, 10^{-3}]$ range. The first interval represents the low values, the second the high values.

The tests are performed by comparison of NMSJ performances and those of SENS-join technique.

5.2. Experimentation results

For low values of selectivity factor, our technique (NMSJ) performs better than SENS-Join. It continues to produce best results for first values of the interval $[10^{-4},10^{-3}]$.

This ascertainment is valid for the two simulation cases: the case of joining three tables and the case of joining five tables (Figs. 3,4,5,6). So, NMSJ keeps the same performances with increasing the number of used tables.

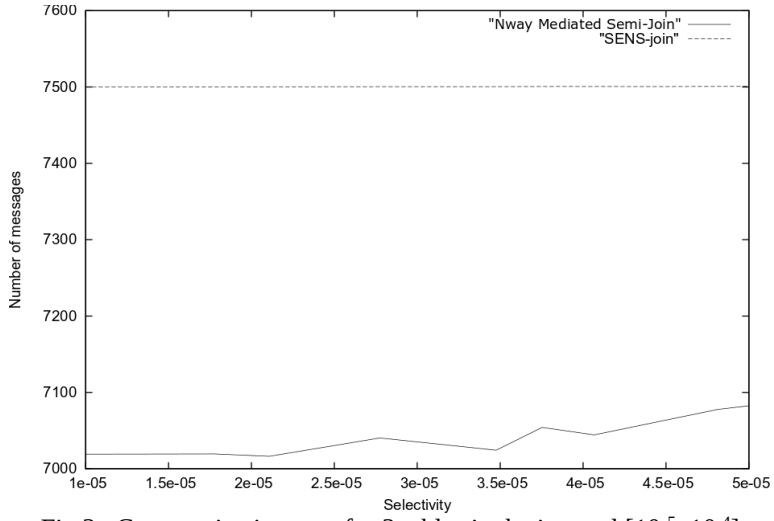

Fig 3. Communication cost for 3 tables in the interval $[10^{-5}, 10^{-4}]$

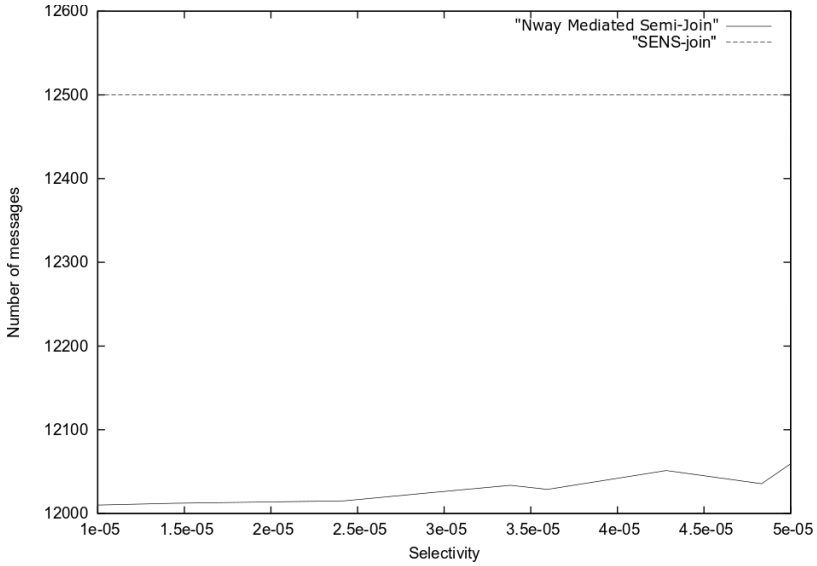

Fig 4. Communication cost for 5 tables in the interval $[10^{-5}, 10^{-4}]$

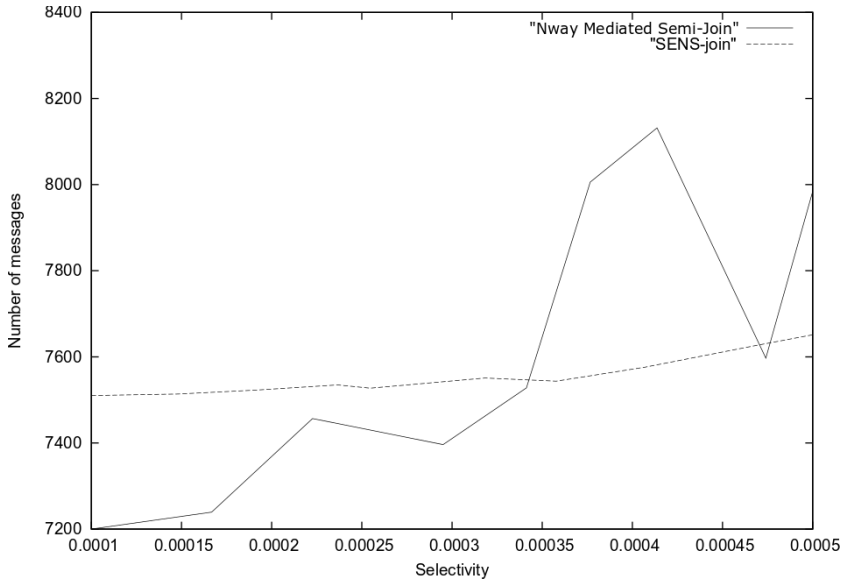

Fig 5. Communication cost for 3 tables in the interval [10^{-4}, 10^{-3}]

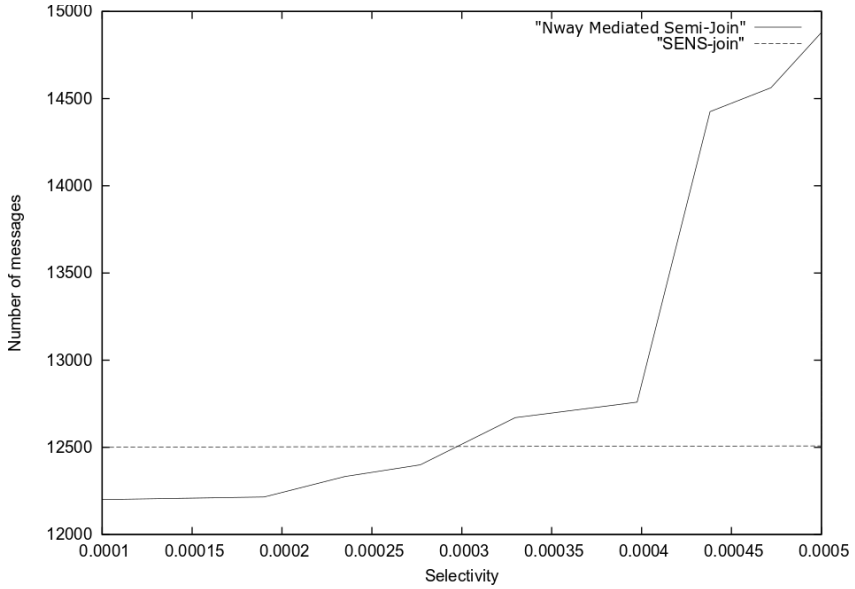

Fig 6. Communication cost for 5 tables in the interval [10^{-4}, 10^{-3}]

5.3. Discussion

NMSJ adopts two interesting principles to reduce considerably the number of transmitted tuples during a query execution. It uses the 'in-network' principle to decrease messages quantity transmitted between internal nodes and to the

sink. Additionally, NMSJ applies the semi-join method to eliminate the non-joinable tuples and so to more minimize volume of transmitted messages. By cons, SENS-join uses only filters established and transmitted by internal nodes to the sink, where the join query will perform. In consequence, NMSJ presents the better results than SENS-join for low values of selectivity factors.

For high values of selectivity factors, NMSJ decreases progressively in performances. This because for high values of selectivity factors, the number of transmitted messages is very high, so it's interesting that the query join executes at the sink.

6. Conclusion

In this paper, we presented NMSJ, a filtering technique to accomplish n-way join queries in wireless sensor networks. NMSJ technique uses two interesting principles to reduce quantity of transmitted messages; it implements the in-network principle and adopts the semi-join approach.

NMSJ shows better performances than the SENS-join with which we compared our technique. NMSJ is better for low values of selectivity factor, but it decreases in performance for the high ones.

In future work, we design to adopt other techniques principles [15] to more improve performances of our technique. Moreover, we will tent to treat joins queries of data streams in wireless sensor networks. Interesting works are those presented in [13] for distributed databases which can be adopted for n-way joins in wireless sensor networks.

On the other hand, we will also extend our solution to treat specific join queries. Recent works in this domain are those of Mo and al. [11] for spatial queries, Kang and al [6] and Lai and al [9] for iceberg joins, and Hasan [5] for parallel joining

R E F E R E N C E S

[1]. *B. J. Bonfils and P. Bonnet*, Adaptive and decentralized operator placement for in-network query processing, Telecommunication Systems, 26, 2004, 389-409.

[2]. *A. Coman, M. A. Nascimento, and J. Sander*, On join location in sensor networks, in 2007 International Conference on Mobile Data Management, 2007, 190-197.

[3]. *B. Djail, K. W. Hidouci, and M. Loudini*, N-way Local SemiJoin : A Filtering Technique for N-Way Joins in Wireless Sensors Networks, Journal of Electronic Systems, 6, 2016, 7-16.

[4]. *B. Djail, W. K. Hidouci, and M. Loudini*, A technique for n-way joins in wireless sensor networks, Database Systems Journal, 7, 2016, 3-9.

[5]. *S. Hasan and A. Amer*, Real-time vehicle ID identification using parallel-joining wireless sensor network, Journal of Fundamental and Applied Sciences, 10, 2018, 663-666.

[6]. *H. Kang*, In-Network Processing of an Iceberg Join Query in Wireless Sensor Networks Based on 2-Way Fragment Semijoins, Sensors, 15, 2015, 6105-6132.

[7]. *H. Kang*, In-network processing of joins in wireless sensor networks, Sensors, 13, 2013, 3358-3393.
[8]. *B. Karp and H.-T. Kung*, GPSR: Greedy perimeter stateless routing for wireless networks, in Proceedings of the 6th annual international conference on Mobile computing and networking, 2000, 243-254.
[9]. *Y. Lai, X. Gao, T. Wang, and Z. Lin*, Efficient iceberg join processing in wireless sensor networks, International Journal of Embedded Systems, 9, 2017, 365-378.
[10]. *J.-K. Min, H. Yang, and C.-W. Chung*, Cost based in-network join strategy in tree routing sensor networks, Information Sciences, 181, 2011, 3443-3458.
[11]. *S. Mo, Y. Fan, Y. Li, and X. Wang*, Multi-attribute join query processing in sensor networks, Journal of Networks, 9, 2014, 2702-2712.
[12]. *M. Steinbrunn, G. Moerkotte, and A. Kemper*, Optimizing join orders: Citeseer, 1993
[13]. *T. M. Tran and B. S. Lee*, Distributed stream join query processing with semijoins, Distributed and Parallel Databases, 27, 2010, 211-254.
[14]. *Y. Yao and J. Gehrke*, Query Processing in Sensor Networks, in CIDR, 2003, 233-244.
[15]. *H. Yu, E.-P. Lim, and J. Zhang*, On in-network synopsis join processing for sensor networks, in 7th International Conference on Mobile Data Management (MDM'06), 2006, 32-32.
[16]. *F. Zhao and L. J. Guibas*, Wireless sensor networks: an information processing approach: Morgan Kaufmann, 2004